

⑩ 日本国特許庁(JP)

⑪ 特許出願公開

⑫ 公開特許公報(A) 平3-237887

⑬ Int. Cl.³

H 04 N 7/133
H 03 M 7/30

識別記号

Z

庁内整理番号

6957-5C
6832-5J

⑭ 公開 平成3年(1991)10月23日

審査請求 未請求 請求項の数 2 (全9頁)

⑮ 発明の名称 DCT処理装置

⑯ 特 願 平2-34310

⑰ 出 願 平2(1990)2月14日

⑱ 発 明 者	藤 原 美 貴 雄	大阪府門真市大字門真1006番地	松下電器産業株式会社内
⑱ 発 明 者	峯 丸 貴 行	大阪府門真市大字門真1006番地	松下電器産業株式会社内
⑱ 発 明 者	高 山 久	大阪府門真市大字門真1006番地	松下電器産業株式会社内
⑲ 出 願 人	松下電器産業株式会社	大阪府門真市大字門真1006番地	
⑳ 代 理 人	弁理士 栗野 重孝	外1名	

明 細 書

1. 発明の名称

DCT処理装置

2. 特許請求の範囲

(1) 画像信号の帯域圧縮で用いられるDCT処理において、Mビット長の信号を $N \times N$ 画素の処理単位でDCT処理を行なう場合に、 $M > N$ の関係が成立する場合、Mビット長を $L < N$ を満足するLビット長の信号に分割し、各Lビット長の部分積の演算をビットシリアルに加算器とROMを用いて演算を実行し、最後にそれらの演算結果を加算することにより、N回のサンプリングクロック期間で、Mビット長の $N \times 1$ の一次元DCT処理を完結することを特徴とするDCT処理装置。

(2) 上記Mビット長の $N \times 1$ の一次元DCT処理装置2個とデータ列のスキャン方向を交換するデュアルポートメモリを用いることを特徴とするMビット長の $N \times N$ の二次元DCT処理装置。

3. 発明の詳細な説明

産業上の利用分野

本発明は、テレビ会議システム、テレビ電話の動画像帯域圧縮でCCITTにより標準化作業がなされている64kビット/秒の画像コーデック処理で用いられるDCT(Discrete Cosine Transform、離散コサイン変換)処理装置に関する。

従来の技術

1画素データがMビット長である $M \times N$ 画素ブロックに対して、DCTを行なう場合、フィルタ処理等の場合と異なり、N画素のデータアクセス期間中に、一次元方向の処理が完結していれば良いという利点がある。この利点を活用して、ビットシリアルに演算処理をおこなう方法が、分散型演算手法として、例えば、アイ・イー・イー・イー・トランザクション・アコースティックス・スピーチ・シグナル・プロセッシング第22巻(1974年12月)第456頁から第462頁(IEEE Trans. Acoustic, Speech, Signal Processing vol. ASSP-22, pp. 456-462, Dec. 1974, "A new hardware realizat

ion of digital filters," by A. Peled and B. Lio
u)に発表されている。この処理手法は M ビット
長のデータに関する演算を、1 ビット目の演算と
いうサブセットに着目して算出し、その結果に対
して2⁽ⁱ⁻¹⁾の桁補正を施して加算することにより
最終結果を求めるというものである。DCT処理
について、この手法を適用すると、以下のように
なる。今、M ビット長で負の数を2の補数で表わ

すN個の整数データ列 $\{u(n) = \sum_{i=0}^{N-1} a_i(n) 2^i, a_{N-1}(n) \in [0, -1], (a_i(n) \in [0, 1], 0 \leq i \leq N-2, 0 \leq n \leq N-1)\}$ に対
する一次元のDCTは、式(1-1)~(1-3)と表現す
ることが出来る。

$$v(k) = \alpha(k) \sum_{n=0}^{N-1} u(n) \cos \left[\frac{\pi(2n+1)k}{2N} \right], 0 \leq k \leq N-1 \quad (1-1)$$

$$\alpha(0) = \left(\frac{1}{N} \right)^{1/2} \quad (1-2)$$

$$\alpha(k) = \left(\frac{2}{N} \right)^{1/2}, 0 \leq k \leq N-1 \quad (1-3)$$

上式(1-1)に、 $u(n)$ の指数表現を代入すると、式(1-4)のように書ける。

$$v(k) = \alpha(k) \sum_{n=0}^{N-1} \sum_{i=0}^{N-1} a_i(n) 2^i \cos \left[\frac{\pi(2n+1)k}{2N} \right], 0 \leq k \leq N-1 \quad (1-4)$$

この式で、1に関する加算でまとめると、次式のようになる。

$$v(k) = \sum_{i=0}^{N-1} 2^i \left(\sum_{n=0}^{N-1} a_i(n) \alpha(k) \cos \left[\frac{\pi(2n+1)k}{2N} \right] \right), 0 \leq k \leq N-1 \quad (1-5)$$

式(1-5)で、大括弧()の中のデータで、 $a_i(n)$ は0か1あるいは0か-1の1ビットのデータであり、 $\alpha(k) \cos \left[\frac{\pi(2n+1)k}{2N} \right]$ はデータ $u(n)$ の値そのものには依存しないので、Nの値が決まれば事前に準備することが可能である。故に、大括弧()

の中の演算は、 $\alpha(k) \cos \left[\frac{\pi(2n+1)k}{2N} \right]$ の値をROM(Read Only Memory)等で準備しておけば、乗算を用いることなく加減算のみで実行することが出来る。集積回路で実現する場合に、並列乗算器を用いる場合に比べてチップサイズを小さくすることが出来る利点を有している。さらにDCTの場合、

会、変換核の $\cos \left[\frac{\pi(2n+1)k}{2N} \right]$ がnにたいして周期(π)で対称性を有することを利用すると、Nが偶数の場合、 $N = 2N'$ として式(1-5)は以下のようにならわすことが出来る。

$$v(k) = \sum_{i=0}^{N-1} 2^i \left\{ \sum_{n=0}^{N-1} a_i(n) \alpha(k) \cos \left[\frac{\pi(2n+1)k}{4N} \right] + \sum_{n=0}^{N-1} a_i(2N'-1-n) \alpha(k) \cos \left[\frac{\pi(2(2N'-1-n)+1)k}{4N} \right] \right\}, 0 \leq k \leq 2N'-1 \quad (1-6)$$

上式の第二項の $\cos(\cdot)$ の項を変形すると

$$\cos \left[\frac{\pi(2(2N'-1-n)+1)k}{4N} \right] = \cos(\pi k) \cos \left[\frac{\pi(2n+1)k}{4N} \right] = -\cos(\pi k) \cos \left[\frac{(2n+1)\pi k}{4N} \right] \quad (1-7)$$

となり、 $K = 2K', 0 \leq k' \leq N'-1$ の時

$$\cos \left[\frac{\pi(2(2N'-1-n)+1)k}{4N} \right] = \cos \left[\frac{(2n+1)\pi k}{4N} \right] \quad (1-8)$$

同様に、 $K = 2K'+1, 0 \leq k' \leq N'-1$ の時

$$\cos \left[\frac{\pi(2(2N'-1-n)+1)k}{4N} \right] = -\cos \left[\frac{(2n+1)\pi k}{4N} \right] \quad (1-9)$$

となる。式(1-8)、(1-9)を用いて、kについて偶数項と奇数項で式(1-7)を変形すると、次式のようになる。

$k = 2K', 0 \leq k' \leq N'-1$ の時

$$v(2k') = \sum_{i=0}^{N-1} 2^i \left\{ \sum_{n=0}^{N-1} (a_i(n) + a_i(2N'-1-n)) \alpha(2k') \cos \left[\frac{2\pi(2n+1)k'}{4N} \right] \right\}, 0 \leq k' \leq N'-1 \quad (1-10)$$

$k = 2k'+1, 0 \leq k' \leq N'-1$ の時

$$v(2k'+1) = \sum_{i=0}^{N-1} 2^i \left\{ \sum_{n=0}^{N-1} (a_i(n) - a_i(2N'-1-n)) \alpha(2k'+1) \cos \left[\frac{\pi(2n+1)(2k'+1)}{4N} \right] \right\}, 0 \leq k' \leq N'-1 \quad (1-11)$$

となる。

式(1-10)と(1-11)により、DCTの変換核 \cos

$\left[\frac{\pi(2n+1)k}{2N} \right]$ の対称性を利用すると、 $\alpha(k) \cos$

$\left[\frac{\pi(2n+1)k}{2N} \right]$ として準備しておくべきROMの容量は、1つのkに対して2^Nワードから2^(N+1)に節約することが出来ることがわかる。しかし、 $(a_1(n) + a_1(2N'-1-n))$ や $(a_1(n) - a_1(2N'-1-n))$ 項からキャリーおよびボロー発生があるので、iに関する加算回数は(k+1)回となる。このように、この演算方式は、大括弧()の中の演算を、DCTの変換

核 $\alpha(k)\cos\left[\frac{\pi(2n+1)k}{2N}\right]$ の値をROM化すると同時にDCTの変換核の対称性を利用してROM容量を節約することができ、演算そのものは乗算を用いることなく加減算のみで実行することが出来る。これらの特徴は、集積回路で実現する場合に、並列乗算器を用いる場合に比べてチップサイズを小さくすることが出来るという利点を有している。

発明が解決しようとする課題

しかしながら、1画素のサンプリング時間が1基本クロック期間であるとして、この1クロック期間に一回の加算処理や一回のROMアクセスが

可能な同期系を想定すると、ビット長Mが、DCTの処理単位Nよりも大きい場合、そのままでは処理が完結しないことを意味する。これは、N-16以上の場合には問題にならないが、CCITTにより標準化作業がなされている84kビット/秒の画像コーデック処理で用いられるN=8のDCTの場合には、 $M \leq 8$ ビットで制限されることになるため、中間処理部で十分な精度を得られないという問題点があった。本発明はかかる点に鑑み、 $M > N$ ビットの精度でNサンプリングクロックの期間で $N \times 1$ の一次元のDCT処理を完結する $N \times N$ のDCT処理装置を安価に提供することを目的とする課題を解決するための手段

上記の問題点を解決するため、本発明のDCT処理装置は、Mビット長を $L < N$ を満足するLビット長に分割し、Lビット長で部分積の演算を並列的に実行し、最後にそれらの中間結果の加算を実行するという構成を備えたものである。

作用

本発明は前記した構成により、Lビット長で部

分積の演算が並列に実行されると、中間和が並列に生成されるために、演算が高速に実行されることとなり、ビット長MがDCTの処理単位Nよりも大きい場合においてもNサンプリングクロックの期間で処理が完結する。

実施例

以下、本発明のDCT処理装置の一実施例を図面と共に説明する。第1図は本発明の一実施例における14ビットの画像信号入力 $u(j)$ に対する 8×1 の一次元のDCT処理装置のブロック図である。図において、2は14ビットの画像信号入力 $u(j)$ 、3~10は14ビットのデータレジスタ、11~18は14ビットの画像信号 $(u(n - \text{mod}(j)), n=0 \sim 7)$ である。19~22はビットシリアル演算部であり、シフトレジスタを用いて、ビットシリアルに加算および減算を行なう。23~38はビットシリアル演算部19~22のビットシリアル演算の結果である各1ビットの信号で、39~42は1ビットの演算結果23~38を各4ビットごとにまとめたデータ線である。43~46はデータ線39~42の4ビットのデータをアドレ

ス情報とし、ROMにより係数とデータの乗算の部分積を生成し、その値に左方シフトを施し累積加算を行なうROMと加算器による係数乗算部である。47~54は 8×1 のDCT処理結果の33ビットの出力信号 $v(k), k=0 \sim 7$ である。55~62は33ビットトライステートドライバであり、出力データの並列/直列変換を行なう。63は33ビットトライステートドライバの55~62の動作により時系列化された33ビット信号出力である。第2図は第1図のビットシリアル演算部19~22の回路構成図である。65は14ビットの画像信号 $u(n)$ 、66は14ビットの画像信号 $u(7-n)$ である。67,68は上位7ビットと下位7ビットが独立な14ビットのデータロード機能付き右方シフターであり、ビットシリアル演算に必要なビット単位での処理を行なう。69,70は1ビット全加算器、71,72は1ビット全減算器である。73~76は1ビットのデータラッチで、1ビット全加算器69,70での演算で発生するキャリーおよび1ビット全減算器71,72での演算で発生するボローを保持する。77~80は各1ビットの演算結果の

信号であり、係数との乗算の部分積をROMから読み出す時のアドレス情報として用いられる。第3図は第1図のROMと加算器による係数乗算部43~48の回路構成図である。82~85は係数との乗算の部分積をROMから読み出す時のアドレス情報である各4ビットのデータである。86~89は16ワード×18ビット容量で、係数との乗算の部分積を生成するROM。90~93は28ビット全加算器。94~97は28ビットのデータロード機能付き右方シフター。98,99は33ビット全加算器。100,101は33ビットレジスタ。102は33ビット出力信号 $v(2k')$ 、103は33ビット出力信号 $v(2k'+1)$ である。第1図と第2図と第3図を用いて、 8×1 の一次元DCT処理の動作について説明する。本発明においては、 $M > 8$ ビット長の1要素データを L ビット長のデータに分割して、処理を実行する。例えば、 M ビット長のデータを J 個の L ビット長データに分割すると、式(1-5)は次のように変形できる。

$$v(k) = \sum_{n=0}^{N-1} 2^{\frac{n-1}{2}} \left(\sum_{k=0}^{N-1} a_1(n) \alpha(k) \cos \left[\frac{2\pi(2n+1)k}{2N} \right] \right)$$

$$\begin{aligned} v(2k') &= \sum_{n=0}^7 2^{\frac{n-1}{2}} \left(\sum_{k=0}^7 (a_1(n) + a_1(7-n)) \alpha(2k') \cos \left[\frac{2\pi(2n+1)k'}{8} \right] \right) \\ &+ 2^{\frac{7}{2}} \sum_{n=0}^7 2^{\frac{n-1}{2}} \left(\sum_{k=0}^7 (a_1(n) + a_1(7-n)) \alpha(2k') \cos \left[\frac{2\pi(2n+1)k'}{8} \right] \right), 0 \leq k' \leq 3 \end{aligned} \quad (1-14)$$

$k=2k'+1, 0 \leq k' \leq 3$ の時

$$\begin{aligned} v(2k'+1) &= \sum_{n=0}^7 2^{\frac{n-1}{2}} \left(\sum_{k=0}^7 (a_1(n) - a_1(7-n)) \alpha(2k'+1) \cos \left[\frac{\pi(2n+1)(2k'+1)}{8} \right] \right) \\ &+ 2^{\frac{7}{2}} \sum_{n=0}^7 2^{\frac{n-1}{2}} \left(\sum_{k=0}^7 (a_1(n) - a_1(7-n)) \alpha(2k') \cos \left[\frac{\pi(2n+1)(2k'+1)}{8} \right] \right), 0 \leq k' \leq 3 \end{aligned} \quad (1-15)$$

M について1回の加算が増加するが、パイプライン構成を適用することにより、 $M=14$ ビットの精度で8要素のサンプリングクロックの期間で 8×1 の一次元のDCT処理を実現することができる。第1図において、 8×1 の一次元DCT処理の動作を説明する。

$$\begin{aligned} &+ \sum_{n=0}^{N-1} 2^{\frac{n-1}{2}} \left(\sum_{k=0}^{N-1} a_1(n) \alpha(k) \cos \left[\frac{\pi(2n+1)k}{2N} \right] \right) \\ &+ \dots \end{aligned}$$

$$\sum_{n=0}^{N-1} 2^{\frac{n-1}{2}} \left(\sum_{k=0}^{N-1} a_1(n) \alpha(k) \cos \left[\frac{\pi(2n+1)k}{2N} \right] \right) \quad (1-12)$$

上式は J 個の部分項の和によって成り立ち、各部分項は L 回の加算により実行されることを意味している。 L 回の加算時間と J 個の項を加算する時間の総和が、 N 個のデータのサンプリング時間よりも短ければ、目的とする高速処理が実現出来る。一例として、 $N=8$ 、 $J=2$ の場合を考える。この時以下の式を満足する M ビット長のデータまで高速処理が可能である。

$8 \geq \text{trunc}(M/2+0.5)+1; \text{trunc}(\cdot)$ 切り捨て(1-13)
故に $M \leq 14$ となる。また、ROM容量削減のため従来例と同様に、式(1-12)に対し式(1-10)、(1-11)を適用すると式(1-14)、(1-15)が得られる。

$k=2k', 0 \leq k' \leq 3$ の時

作を説明する。14ビットの画像信号入力 $u(j)$ は8要素のサブセットに対してDCT処理を施されるため、14ビットレジスタ3~10に、それぞれ $(u(n), n=\text{mod}(j), 0 \leq n \leq 7)$ と分割されて保持される。14ビットレジスタ3~10では、この8個のサブセットデータ列 $\{u(n), 0 \leq n \leq 7\}$ が完全に更新されるまで、1回のデータサンプリングに対して1回のシフト動作を行ない、データを順次送っていく。つまり、8回のデータサンプリング毎に、新しいサブセットデータが14ビットレジスタ3~10に $u(7), \dots, u(0)$ としてセットされる。次に、このデータは14ビットの信号線11~18を介して、それぞれビットシリアル演算部19~22に供給される。このビットシリアル演算部19~22における処理を、第2図を用いて説明する。14ビットの画像入力65~68は、第1図の14ビットレジスタ3~10のいずれかからのデータで、2の補数表現を用いて現わすと、 $u(n) = \sum_{i=0}^{13} a_i(n) 2^i (a_i(n) \in [0, -1], a_i(n) \in [0, 1], 0 \leq i \leq 12, 0 \leq n \leq 3)$ と、 $u(7-n) = \sum_{i=0}^{13} a_i(7-n) 2^i$

$(a_i(n) \in [0, -1], a_i(7-n) \in [0, 1], 0 \leq i \leq 12, 0 \leq n \leq 3)$ である。これらのデータが 上位7ビットと下位7ビットが独立した14ビットのデータロード機能付き右方シフター87,88に入力され、それぞれ $u(n) = \sum_{i=0}^6 a_i(n)2^i + 2^7 \sum_{i=0}^6 a_{i+7}(n)2^i$, $u(7-n) = \sum_{i=0}^6 a_i(7-n)2^i + 2^7 \sum_{i=0}^6 a_{i+7}(7-n)2^i$ として上位7ビットと下位7ビットが分離した形で処理され、1クロック期間毎に1回のLSB側への右方シフトが実行される。データロード機能付き右方シフター87,88より出力される信号は $u(n)$ および $u(7-n)$ の上位7ビットと下位7ビットに関して2'桁の各1ビットの値で、 $a_i(n)$ と $a_{i+7}(n)$ と $a_i(7-n)$ と $a_{i+7}(7-n)$ である。これらの信号により、1ビット全加算器69,70と1ビット全減算器71,72において、式(1-14)、(1-15)の右辺の $(a_{i+7}(n)+a_i(7-n))$, $(a_i(n)+a_{i+7}(7-n))$, $((a_{i+7}(n)-a_i(7-n)))$, $((a_i(n)-a_{i+7}(7-n)))$ の演算を実行する。これらの演算により発生するキャリーおよびボローは1ビットラッチ73~76に保持され、1クロック後の演算に用いられるた

めに、元の1ビット全加算器69,70と1ビット全減算器71,72に再帰的に入力される。1ビット全加算器69,70の演算結果は1ビットデータ線77,78に各々出力され、1ビット全減算器71,72の演算結果は1ビットデータ線79,80に各々出力される。

第2図で説明したのと同様に、ビットシリアル演算部19,22では、式(1-4)、(1-5)の右辺の $(a_{i+7}(n)+a_i(7-n))$, $(a_i(n)+a_{i+7}(7-n))$, $((a_{i+7}(n)-a_i(7-n)))$, $((a_i(n)-a_{i+7}(7-n)))$ の演算が実行され、ビットシリアル演算部19では $u(0)$ と $u(7)$ について、ビットシリアル演算部20では $u(1)$ と $u(6)$ について、ビットシリアル演算部21では $u(2)$ と $u(5)$ について、ビットシリアル演算部22では $u(3)$ と $u(4)$ について、この演算を実行する。この結果、各ビットシリアル演算部19~22より出力される4ビットデータ線39~42は、4ビットデータ線39が $((a_{i+7}(n)+a_i(7-n))$, $n=0,1,2,3)$ を示し、4ビットデータ線40が $((a_i(n)+a_{i+7}(7-n))$, $n=0,1,2,3)$ を示し、4ビットデータ線41が $((a_{i+7}(n)-a_i(7-n))$, $n=0,1,2,3)$ を示し、4ビットデータ線42が $((a_i(n)-a_{i+7}(7-n))$, $n=0,1,2,3)$ を示している。これらの4ビットの信号の意味を、もう少し詳しく説明するために、式(1-14)、(1-15)に戻って説明する。式(1-14)および式(1-15)のnに関する和の部分を展開すると、次式のように表現することが出来る。

$k=2k', 0 \leq k' \leq 3$ の時

$$v(2k') = \sum_{i=0}^6 2^i ((a_i(0)+a_i(7)) \alpha(2k') \cos [\frac{2\pi k'}{8}])$$

$$+ (a_i(1)+a_i(6)) \alpha(2k') \cos [\frac{6\pi k'}{8}]$$

$$+ (a_i(2)+a_i(5)) \alpha(2k') \cos [\frac{10\pi k'}{8}]$$

$$+ (a_i(3)+a_i(4)) \alpha(2k') \cos [\frac{14\pi k'}{8}]$$

$$+ 2^7 \sum_{i=0}^6 2^i ((a_{i+7}(0)+a_{i+7}(7)) \alpha(2k') \cos [\frac{2\pi k'}{8}])$$

$$+ (a_{i+7}(1)+a_{i+7}(6)) \alpha(2k') \cos [\frac{6\pi k'}{8}]$$

$$+ (a_{i+7}(2)+a_{i+7}(5)) \alpha(2k') \cos [\frac{10\pi k'}{8}]$$

$$+ (a_{i+7}(3)+a_{i+7}(4)) \alpha(2k') \cos [\frac{14\pi k'}{8}]$$

$$, 0 \leq k' \leq 3 \quad (1-16)$$

$k=2k'+1, 0 \leq k' \leq 3$ の時

$$v(2k'+1) = \sum_{i=0}^6 2^i ((a_i(0)-a_i(7)) \alpha(2k'+1) \cos [\frac{\pi(2k'+1)}{8}])$$

$$+ (a_i(1)-a_i(6)) \alpha(2k'+1) \cos [\frac{3\pi(2k'+1)}{8}]$$

$$+ (a_i(2)-a_i(5)) \alpha(2k'+1) \cos [\frac{5\pi(2k'+1)}{8}]$$

$$+ (a_i(3)-a_i(4)) \alpha(2k'+1) \cos [\frac{7\pi(2k'+1)}{8}]$$

$$+ 2^7 \sum_{i=0}^6 2^i ((a_{i+7}(0)-a_{i+7}(7)) \alpha(2k'+1) \cos [\frac{\pi(2k'+1)}{8}])$$

$$+ (a_{i+7}(1)-a_{i+7}(6)) \alpha(2k'+1) \cos [\frac{3\pi(2k'+1)}{8}]$$

$$+ (a_{i+7}(2)-a_{i+7}(5)) \alpha(2k'+1) \cos [\frac{5\pi(2k'+1)}{8}]$$

$$+ (a_{i+7}(3)-a_{i+7}(4)) \alpha(2k'+1) \cos [\frac{7\pi(2k'+1)}{8}]$$

$$, 0 \leq k' \leq 3 \quad (1-17)$$

このように、上式(1-16)における各2'桁に関する演算は、 K' を固定すれば、 $((a_{i+7}(n)+a_i(7-n))$, $n=0,1,2,3)$ の4ビットのデータと $((a_i(n)+a_{i+7}(7-n))$, $n=0,1,2,3)$ の4ビットのデータによって

一意的に決定することが出来る。又、式(1-17)についても同様のことが成立する。故にこれらの4ビット信号をアドレス情報とし、そのアドレス情報に従い $\left\{\sum_{n=0}^7 (a_{1+r}(n)+a_{1+r}(7-n))\alpha(2k')\cos\left[\frac{2\pi(2n+1)k'}{8}\right]\right\}$ を出力するようにROM化することは容易である。このように、4ビットデータ線39の4ビットデータは、式(1-14)における2'桁での $\left\{\sum_{n=0}^7 (a_{1+r}(n)+a_{1+r}(7-n))\alpha(2k')\cos\left[\frac{2\pi(2n+1)k'}{8}\right]\right\}$ を求めるアドレス情報として用いられ、ROMと加算器による係数乗算部43~46に入力される。同様に、4ビットデータ線40の4ビットデータは、式(1-14)における2'桁での $\left\{\sum_{n=0}^7 (a_{1+r}(n)+a_{1+r}(7-n))\alpha(2k')\cos\left[\frac{2\pi(2n+1)k'}{8}\right]\right\}$ を求めるアドレス情報として用いられ、ROMと加算器による係数乗算部43~46に入力される。同様に、4ビットデータ線41の4ビットデータは、式

式(1-14)における2'桁での $\left\{\sum_{n=0}^7 (a_{1+r}(n)+a_{1+r}(7-n))\alpha(2k')\cos\left[\frac{2\pi(2n+1)k'}{8}\right]\right\}$ を求めるアドレス情報で、4ビットデータ線40を介して入力される。同様に、4ビット信号84は、式(1-15)における2'桁での $\left\{\sum_{n=0}^7 (a_{1+r}(n)-a_{1+r}(7-n))\alpha(2k')\cos\left[\frac{\pi(2n+1)(2k'+1)}{8}\right]\right\}$ を求めるアドレス情報で、4ビットデータ線41を介して入力される。同様に、4ビット信号85は、式(1-15)における2'桁での $\left\{\sum_{n=0}^7 (a_{1+r}(n)-a_{1+r}(7-n))\alpha(2k')\cos\left[\frac{\pi(2n+1)(2k'+1)}{8}\right]\right\}$ を求めるアドレス情報で、4ビットデータ線42を介して入力される。次に、16ワード×18ビット容量のROM86では、4ビット信号82をアドレス情報として受け $\left\{\sum_{n=0}^7 (a_{1+r}(n)+a_{1+r}(7-n))\alpha(2k')\cos\left[\frac{2\pi(2n+1)k'}{8}\right]\right\}$ の値を18ビットのデータとして出力する。同様に、

(1-15)における2'桁での $\left\{\sum_{n=0}^7 (a_{1+r}(n)-a_{1+r}(7-n))\alpha(2k')\cos\left[\frac{\pi(2n+1)(2k'+1)}{8}\right]\right\}$ を求めるアドレス情報として用いられ、ROMと加算器による係数乗算部43~46に入力される。同様に、4ビットデータ線42の4ビットデータは、式(1-15)における2'桁での $\left\{\sum_{n=0}^7 (a_{1+r}(n)-a_{1+r}(7-n))\alpha(2k')\cos\left[\frac{\pi(2n+1)(2k'+1)}{8}\right]\right\}$ を求めるアドレス情報として用いられ、ROMと加算器による係数乗算部43~46に入力される。次に、ROMと加算器による係数乗算部43~46の中での処理について、第3図を用いて説明する。第3図において、4ビット信号82は、式(1-14)における2'桁での $\left\{\sum_{n=0}^7 (a_{1+r}(n)+a_{1+r}(7-n))\alpha(2k')\cos\left[\frac{2\pi(2n+1)k'}{8}\right]\right\}$ を求めるアドレス情報で、4ビットデータ線39を介して入力される。同様に、4ビット信号線83は、

16ワード×18ビット容量のROM87では、4ビット信号83をアドレス情報として受け $\left\{\sum_{n=0}^7 (a_{1+r}(n)+a_{1+r}(7-n))\alpha(2k')\cos\left[\frac{2\pi(2n+1)k'}{8}\right]\right\}$ の値を18ビットのデータとして出力する。同様に、16ワード×18ビット容量のROM88では、4ビット信号84をアドレス情報として受け $\left\{\sum_{n=0}^7 (a_{1+r}(n)-a_{1+r}(7-n))\alpha(2k')\cos\left[\frac{\pi(2n+1)(2k'+1)}{8}\right]\right\}$ の値を18ビットのデータとして出力する。同様に、16ワード×18ビット容量のROM89では、4ビット信号85をアドレス情報として受け $\left\{\sum_{n=0}^7 (a_{1+r}(n)-a_{1+r}(7-n))\alpha(2k')\cos\left[\frac{\pi(2n+1)(2k'+1)}{8}\right]\right\}$ の値を18ビットのデータとして出力する。次に26ビット全加算器90~93と、26ビットのデータロード機能付き右方シフター94~97は、4組の26ビット累積加算器として働き、前記ROM86~89からの18ビットの出力データは、26ビット全加算器90~93の一方の入力のMSB側18ビットに入力される。26ビット全加算器90~93で

の加算結果は、それぞれ28ビットのデータロード機能付き右方シフター94~97でLSB側に(右方に)1ビットシフトされ、次のクロック期間で、前記ROM86~89の出力と加算される。但し、この動作で、 $i=0$ の時には、28ビットのデータロード機能付きシフター94~97から28ビット全加算器90~93に輸入されるデータは'0'に初期化される。この操作により、8回のクロック期間で、式(1-14)、(1-15)のそれぞれ4つの項が算出される。33ビット全加算器98~99では、28ビットシフター94~97の出力を加算する。ここで、28ビットシフター94と96の出力は加算時に 2^i で桁補正が行なわれ、式(1-14)、(1-15)の $v(2k')$ 、 $(2k'+1)$ の値を算出する。そして、33ビットレジスタ100、101に、その演算結果をセットする。33ビットレジスタ100、101は次の8クロックの期間、新しいサブセットに対して $v(2k')$ 、 $(2k'+1)$ の値が算出されるまで、現在の値を保持する。ここで第1図に戻って、説明を続ける。第3図における前記33ビットレジスタ100、101からのデータ102、103は、第1図の47~

54に対応し、他の3つのブロックの信号の出力信号と合わせて、DCT処理された信号列 $(v(k), 0 \leq k \leq 7)$ となる。この33ビット出力信号列 $(v(k), 0 \leq k \leq 7)$ がそれぞれトライステートドライバ55~62により、時系列化されて出力端子63より出力される。第4図は本発明の一実施例によるアダプティブDCT処理装置の概略構成を示すものである。104は制御信号入力端子、105はデータストロープ信号入力端子、106は14ビットの画像信号入力端子、107は14ビットの参照画像信号入力端子、108は差分器、109はクリッピング回路、110は 8×1 の一次元のDCT処理回路111に対するタイミング信号生成回路、112はクリッピング・丸め込み処理回路、113は128ワード \times 16ビットのデュアルポートメモリ114への書き込み制御回路、115はデュアルポートメモリ114からの読み出し制御回路、116は 8×1 の一次元のDCT処理回路117に対するタイミング信号生成回路、118はクリッピング・丸め込み処理回路、119は14ビットの画像出力端子である。第4図は第1図の 8×1 のDCT処理回路

ブロックを利用した 8×8 のアダプティブDCT処理装置の一例である。制御信号104によりアダプティブ処理を行なう場合は、差分器108において、14ビット画像信号入力106と、14ビット参照画像信号入力107の差分をとる。その結果の信号が前提とされている最大・最小のしきい値を超える場合は、クリッピング回路109でクリップされ、 8×1 の一次元のDCT処理回路111に輸入される。クリップを行なわない場合は、差分器108からの信号がスルーされ、 8×1 の一次元のDCT処理回路111に輸入され、 8×1 のDCT処理が施される。 8×1 の一次元のDCT処理回路111における処理タイミングは、データストロープ信号入力端子105より入力される14ビット画像信号入力106から入力される一組64個のデジタル画像信号の先頭の信号を示すストロープ信号をトリガーとして、タイミング信号生成回路110により制御される。次に、クリッピング・丸め込み処理回路112では、 8×1 の一次元のDCT処理回路111からの処理出力に対しクリッピング・丸め込み処理を行ない、その結果を128ワ

ード \times 16ビットのデュアルポートメモリ114に輸入する。128ワード \times 16ビットのデュアルポートメモリ114の書き込み、読みだしは、書き込み制御回路113、読みだし制御回路115により制御される。次に、 8×1 の一次元のDCT処理回路117では、128ワード \times 16ビットのデュアルポートメモリ114からの入力信号をDCT処理し、ここでの処理タイミングはタイミング信号生成回路116により制御される。 8×1 の一次元のDCT処理回路117からの出力データは、クリッピング・丸め込み処理回路118を通じて、14ビットの画像出力端子119に出力され、二次元の 8×8 のDCT処理が完結する。なお本実施例では1画素データが14ビット長の時、7ビット長の信号に2分割したが、 $M > N$ を満たすMビット長をLビット長の信号に分割しても(ただしL1ビット長の信号に分割する場合を除く)同様の効果を有する。

発明の効果

以上説明したごとく本発明によれば、Mビット長を $L < N$ を満足するLビット長に分割し、L

ビット長で部分積の演算を並列的に実行し、最後にそれらの中間結果の加算を実行する方式により、 $N=8, J=2$ の時に、8つのサンプリングクロックの期間で 8×1 の一次元のDCT処理を実現することができ、かつ内部演算精度を $M=14$ ビットの精度まで乗算器を用いずに確保することができ、その実用的効果は大きい。

4. 図面の簡単な説明

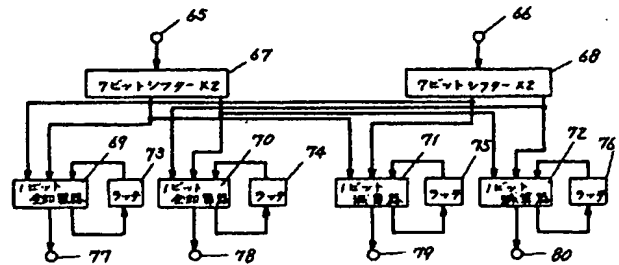
第1図は本発明の一実施例における 8×1 の一次元のDCT処理回路のブロック図、第2図はビットシリアル演算部の回路構成図、第3図はROMと加算器による係数乗算部の回路構成図、第4図は本発明の一実施例によるアダプティブDCT処理回路の概略構成図である。

2……画像信号入力 3～10……データレジスタ、19～22……ビットシリアル演算部 43～46……ROMと加算器による係数乗算部 111, 117…… 8×1 の一次元DCT処理回路 114……デュアルポートメモリ。

代理人の氏名 弁理士 栗野重孝 ほか1名

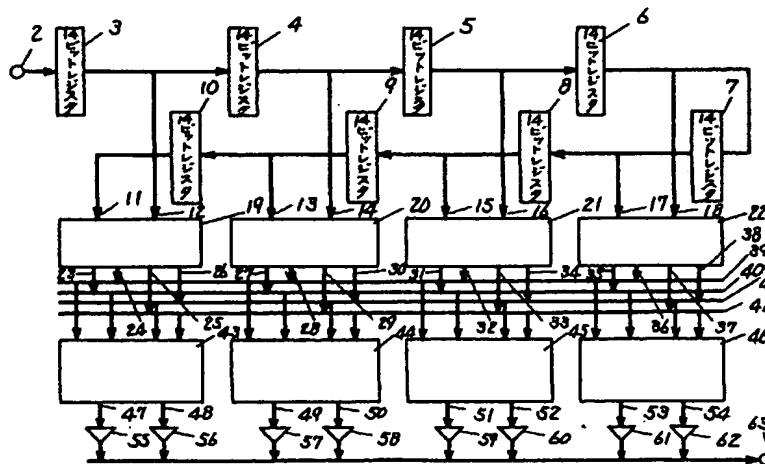
67, 68 …… 14ビットのゲートロッド駆動付き右方シフター
69, 70 …… 1ビット全加算器
71, 72 …… 1ビット全減算器
73, 74 …… 1ビットデータラッチ

第2図

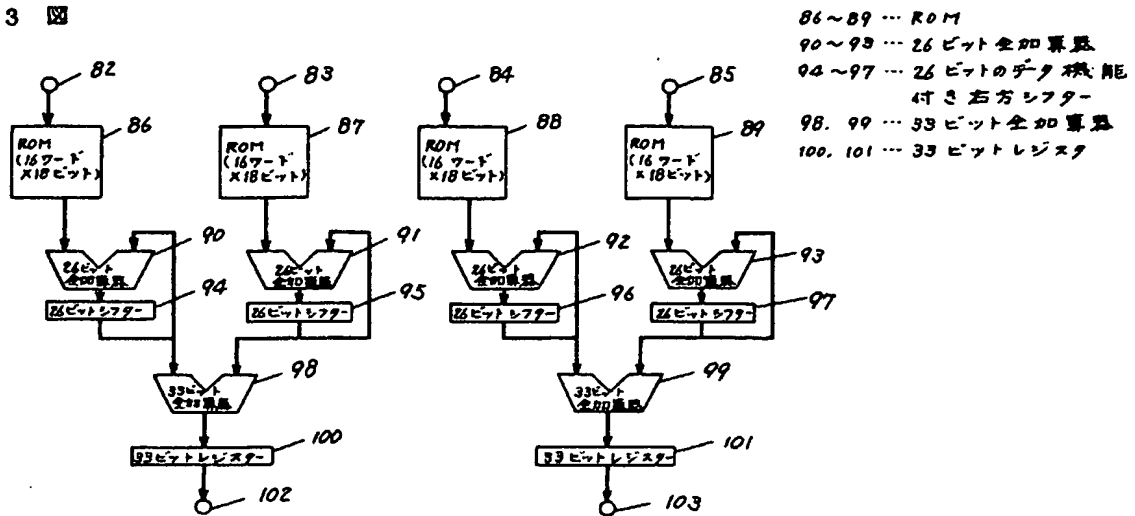


2 …… 14ビットの画像信号入力
3～10 …… 14ビットのデータレジスタ
11～18 …… 14ビットの画像信号
19～22 …… ビットシリアル演算部
43～46 …… ROMと加算器による係数乗算部
55～62 …… トライスタートドライバ

第1図



第 3 図



108 ... 差分器
 109 ... クリッピング回路
 110 ... タイミング信号生成回路
 111, 117 ... 8×1の一次元DCT処理回路
 112, 118 ... クリッピング後の込み処理回路
 113 ... 書き込み制御回路
 114 ... デュアルポートメモリ
 115 ... 読み出し制御回路

第 4 図

